CENTER FOR BIOLOGICAL SEQUENCE ANALYSIS

Multiple Alignment

Anders Gorm Pedersen
Molecular Evolution Group
Center for Biological Sequence Analysis
gorm@cbs.dtu.dk

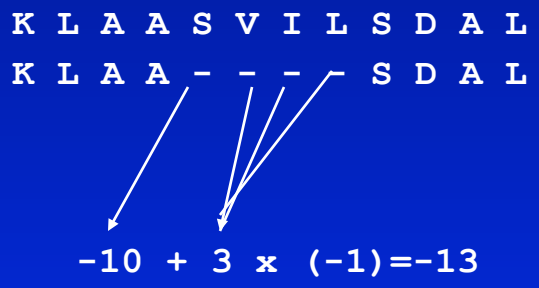
Refresher: pairwise alignments


```
43.2% identity;                    Global alignment score: 374

                10        20        30        40                50
alpha   V-LSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHF-DLS-----HGSA
        : :.: .:. : : ::::  .. : :.::: :... .: :. .:  : :::       :.
beta    VHLTPEEKSAVTALWGKV--NVDEVGGEALGRLLVVYPWTQRFFESFGDLSTPDAVMGNP
               10          20        30        40        50

             60        70        80        90       100       110
alpha   QVKGHGKKVADALTNAVAHVDDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHL
        .::.::::  :.....::.:. .....::.::  ::.::: ::.::.. :. .:: :.
beta    KVKAHGKKVLGAFSDGLAHLDNLKGTFATLSELHCDKLHVDPENFRLLGNVLVCVLAHHF
        60        70        80        90       100       110

           120       130       140
alpha   PAEFTPAVHASLDKFLASVSTVLTSKYR
          :::: :.:. .: .:.:...:. ::.
beta    GKEFTPPVQAAYQKVVAGVANALAHKYH
      120       130       140
```

Refresher: pairwise alignments


```
A   5
R  -2  7
N  -1 -1  7
D  -2 -2  2  8
C  -1 -4 -2 -4 13
Q  -1  1  0  0 -3  7
E  -1  0  0  2 -3  2  6
G   0 -3  0 -1 -3 -2 -3  8
.
.
.
    A  R  N  D  C  Q  E  G ...
```

- Alignment score is calculated from substitution matrix
- Identities on diagonal have high scores
- Similar amino acids have high scores
- Dissimilar amino acids have low (negative) scores



- Gaps penalized by gap-opening + gap elongation

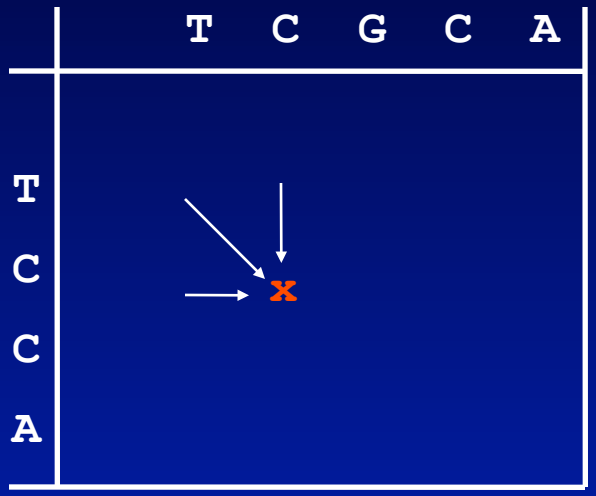
Refresher: pairwise alignments

The number of possible pairwise alignments increases explosively with the length of the sequences:

Two protein sequences of length 100 amino acids can be aligned in approximately 10^{60} different ways

10^{60} bottles of beer would fill up our entire galaxy

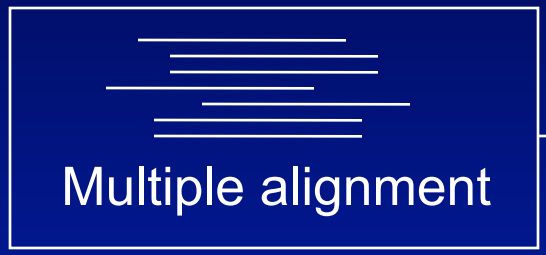
Refresher: pairwise alignments



- Solution:
  dynamic programming
- Essentially:
  the best path through any grid point in the alignment matrix must originate from one of three previous points
- Far fewer computations
- Best alignment guaranteed to be found

Refresher: pairwise alignments

- Most used substitution matrices are themselves derived empirically from simple multiple alignments



Multiple alignment

Calculate substitution frequencies

```
A/A   2.15%
A/C   0.03%
A/D   0.07%
...
```

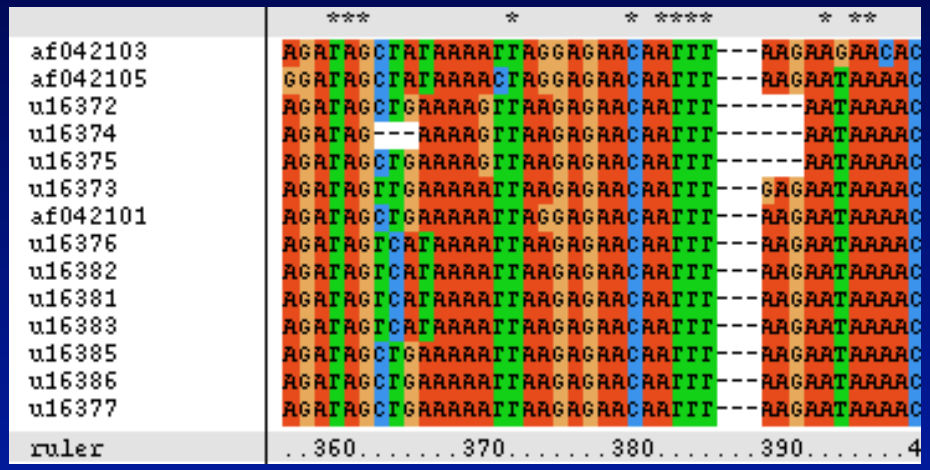
Convert to scores

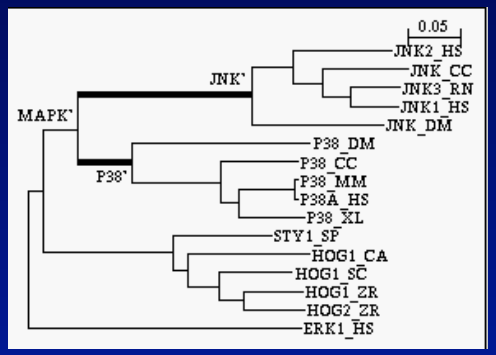
$$\text{Score}(A/C) = \log \frac{\text{Freq}(A/C),\text{observed}}{\text{Freq}(A/C),\text{expected}}$$

Multiple alignment

Multiple alignments: what use are they?

- Starting point for studies of molecular evolution

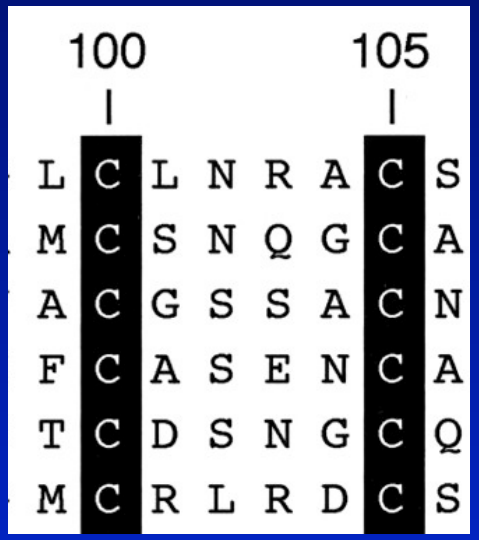


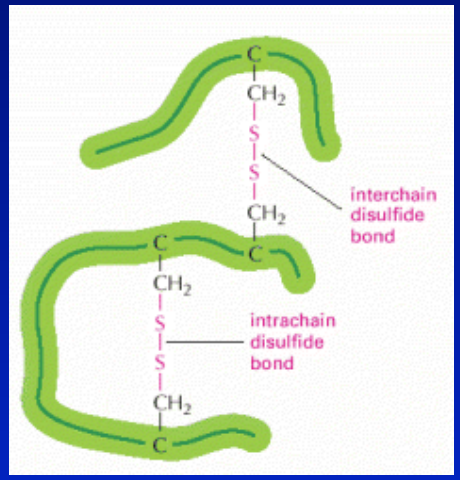


CENTER FOR BIOLOGICAL SEQUENCE ANALYSIS

Multiple alignments: what use are they?

- Characterization of protein families:
  - Identification of conserved (functionally important) sequence regions
  - Construction of profiles for further database searching
  - Prediction of structural features (disulfide bonds, amphipathic alpha-helices, surface loops, etc.)





Scoring a multiple alignment: the “sum of pairs” score

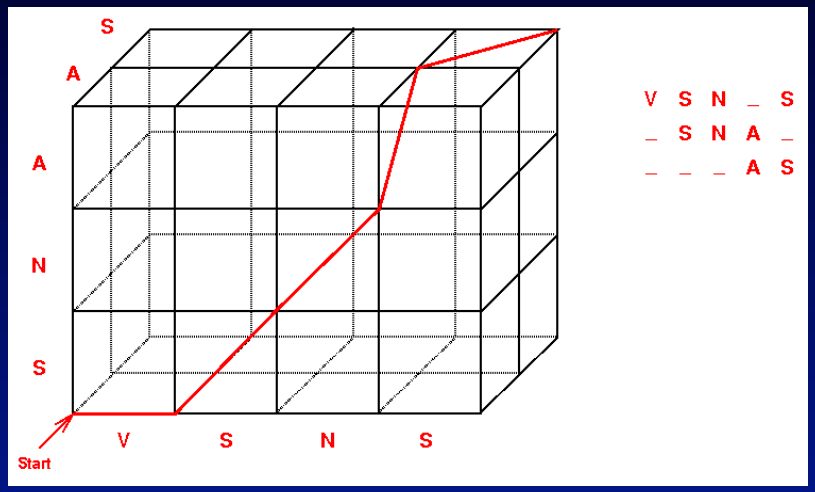
One column
from alignment

```
AA: 4, AS: 1, AT:0
AS: 1, AT: 0
ST: 1

SP- score: 4+1+0+1+0+1 = 7
```

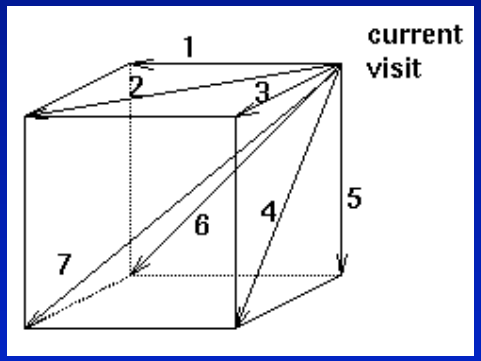
Weighted sum of pairs: each SP-score is multiplied by a weight reflecting the evolutionary distance (avoids undue influence on score by sets of very similar sequences)

=> In theory, it is possible to define an alignment score
for multiple alignments (there are several alternative scoring systems)

Multiple alignment: dynamic programming is only feasible for very small data sets



Dynamic programming matrix for 3 sequences



For 3 sequences, optimal path must come from one of 7 previous points

- In theory, optimal multiple alignment can be found by dynamic programming using a matrix with more dimensions (one dimension per sequence)
- BUT even with dynamic programming finding the optimal alignment very quickly becomes impossible due to the astronomical number of computations
- Full dynamic programming only possible for up to about 4-5 protein sequences of average length
- Even with heuristics, not feasible for more than 7-8 protein sequences
- Never used in practice

Multiple alignment: an approximate solution

- Progressive alignment (ClustalX and other programs):

1. Perform all *pairwise* alignments; keep track of sequence similarities between all pairs of sequences (construct “distance matrix”)

2. Align the most similar pair of sequences

3. Progressively add sequences to the (constantly growing) multiple alignment in order of decreasing similarity.

Progressive alignment: details

1) Perform all pairwise alignments, note pairwise distances (construct “distance matrix”)

6 pairwise alignments

	S1	S2	S3	S4
S1				
S2	3			
S3	1	3		
S4	3	2	3	

2) Construct pseudo-phylogenetic tree from pairwise distances

	S1	S2	S3	S4
S1				
S2	3			
S3	1	3		
S4	3	2	3	

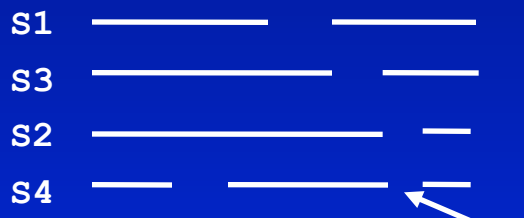
“Guide tree”

Progressive alignment: details

3) Use tree as guide for multiple alignment:
   a) Align most similar pair of sequences using dynamic programming

   b) Align next most similar pair

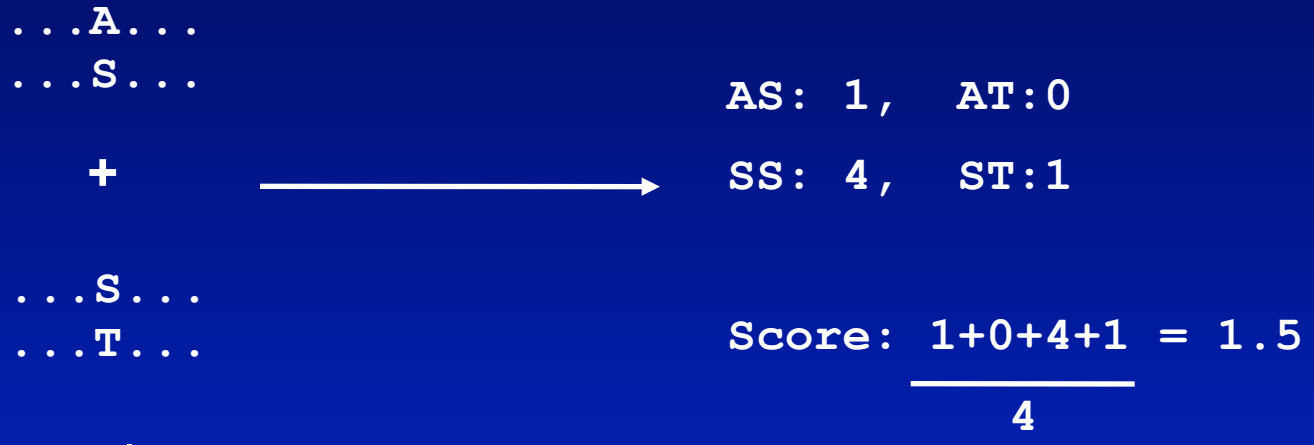
   c) Align alignments using dynamic programming - preserve gaps



New gap to optimize alignment of (S2,S4) with (S1,S3)

Scoring profile alignments

Compare each residue in one profile to all residues in second profile. Score is average of all comparisons.



One column
from alignment

Additional ClustalX heuristics

- Sequence weighting:
  - scores from similar groups of sequences are down-weighted
- Variable substitution matrices:
  - during alignment ClustalX uses different substitution matrices depending on how similar the sequences/profiles are
- Variable gap penalties:
  - gap penalties depend on substitution matrix
  - gap penalties depend on similarity of sequences
  - reduced gap penalties at existing gaps
  - increased gap penalties CLOSE to existing gaps
  - reduced gap penalties in hydrophilic stretches (presumed surface loop)
  - residue-specific gap penalties
  - and more...

Other multiple alignment programs

pileup
multalign
multal
saga
hmmt
MUSCLE
ProbCons

DIALIGN
SBpima
MLpima
T-Coffee
mafft
poa
prank
...

Quantifying the Performance of Protein Sequence Multiple Alignment Programs

- Compare to alignment that is known (or strongly believed) to be correct
- Quantify by counting e.g. fraction of correctly paired residues
- Option 1: Compare performance to benchmark data sets for which 3D structures and structural alignments are available (BALiBASE, PREfab, SABmark, SMART).
  - Advantage: real, biological data with real characteristics
  - Problem: we only have good benchmark data for core regions, no good knowledge of how gappy regions really look
- Option 2: Construct synthetic alignments by letting a computer simulate evolution of a sequence along a phylogenetic tree
  - Advantage: we know the real alignment including where the gaps are
  - Problem: Simulated data may miss important aspects of real biological data

Performance on BALiBASE benchmark

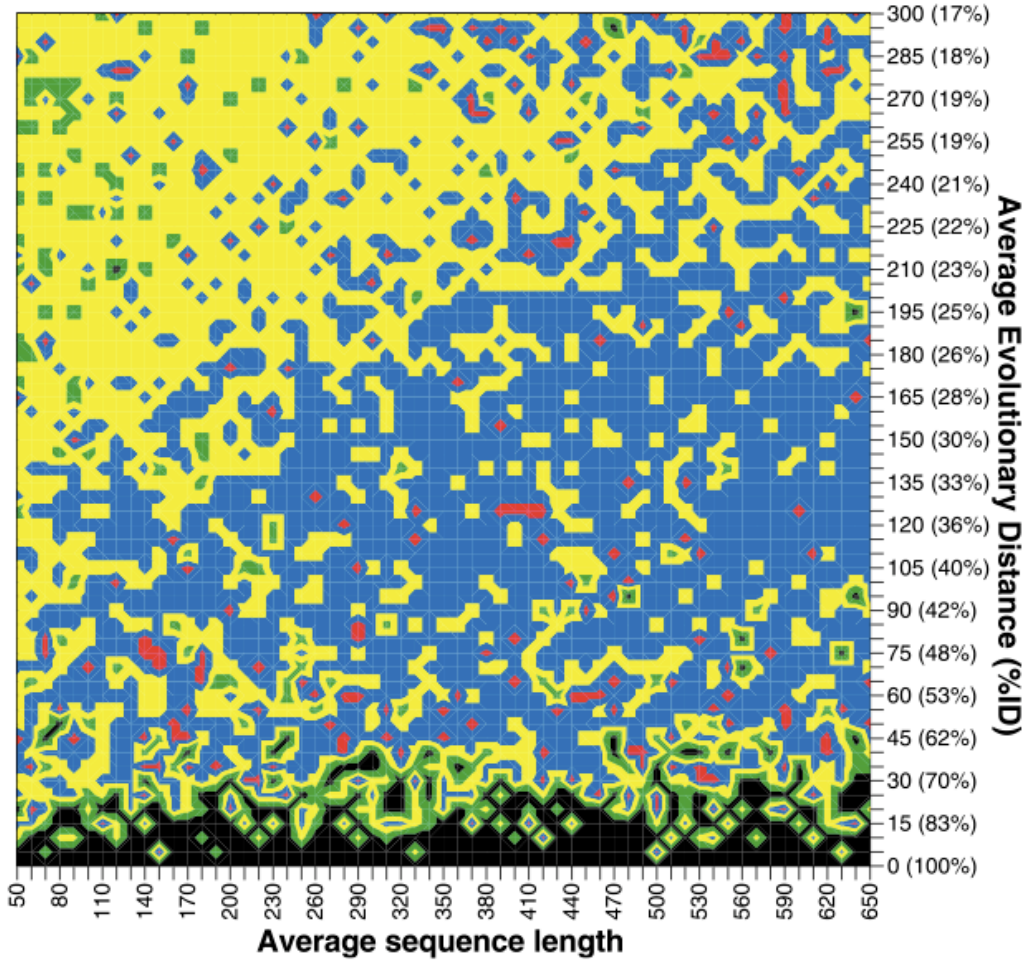


Fig. 1. Color coded matrix showing which method performed best for each pair-combination of conditions: average sequence length (x-axis) and average evolutionary distance (y-axis). The methods are Poa (green), Dialign (yellow), T-Coffee (blue) and ClustalW (red).

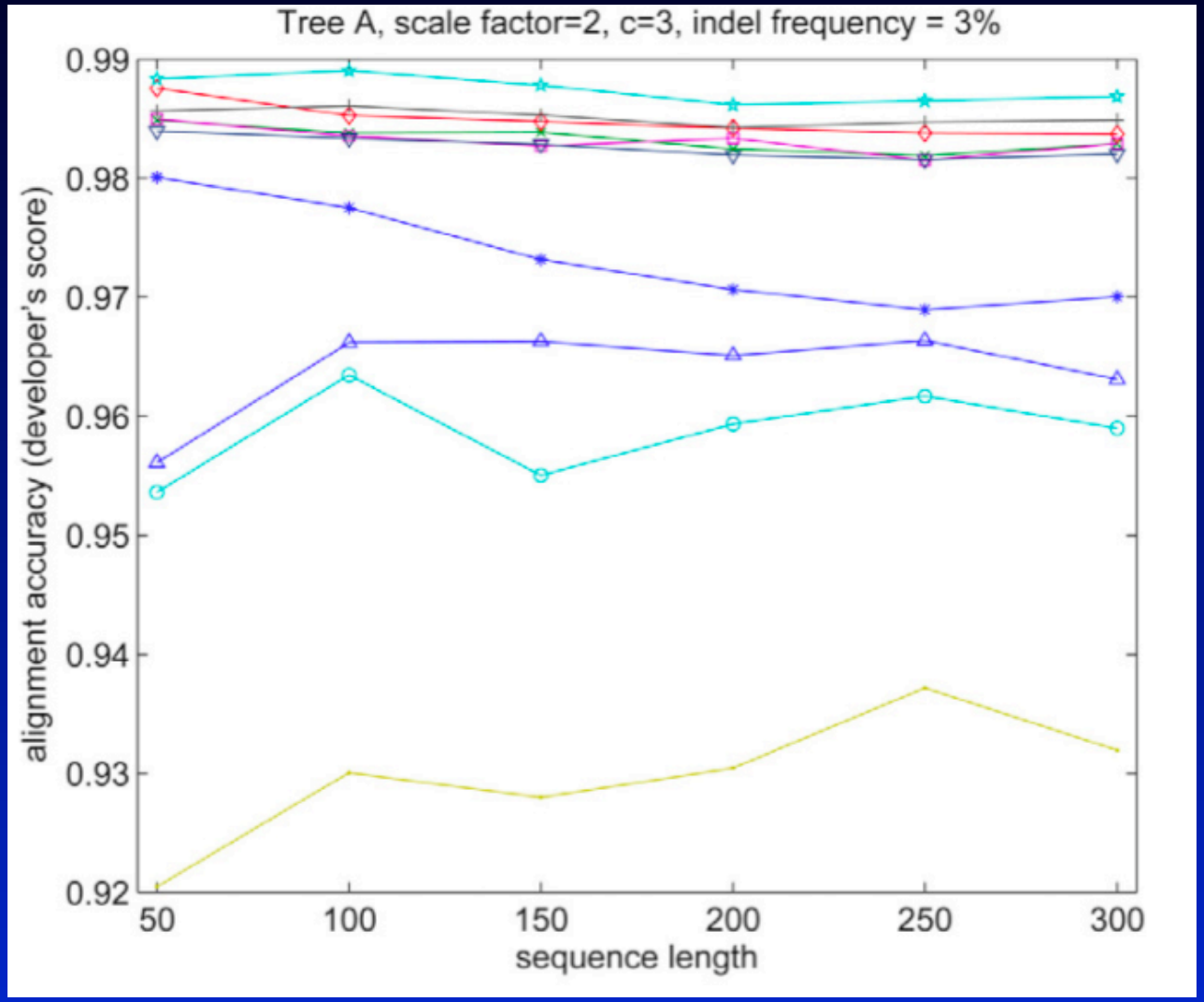
Dialign

T-Coffee

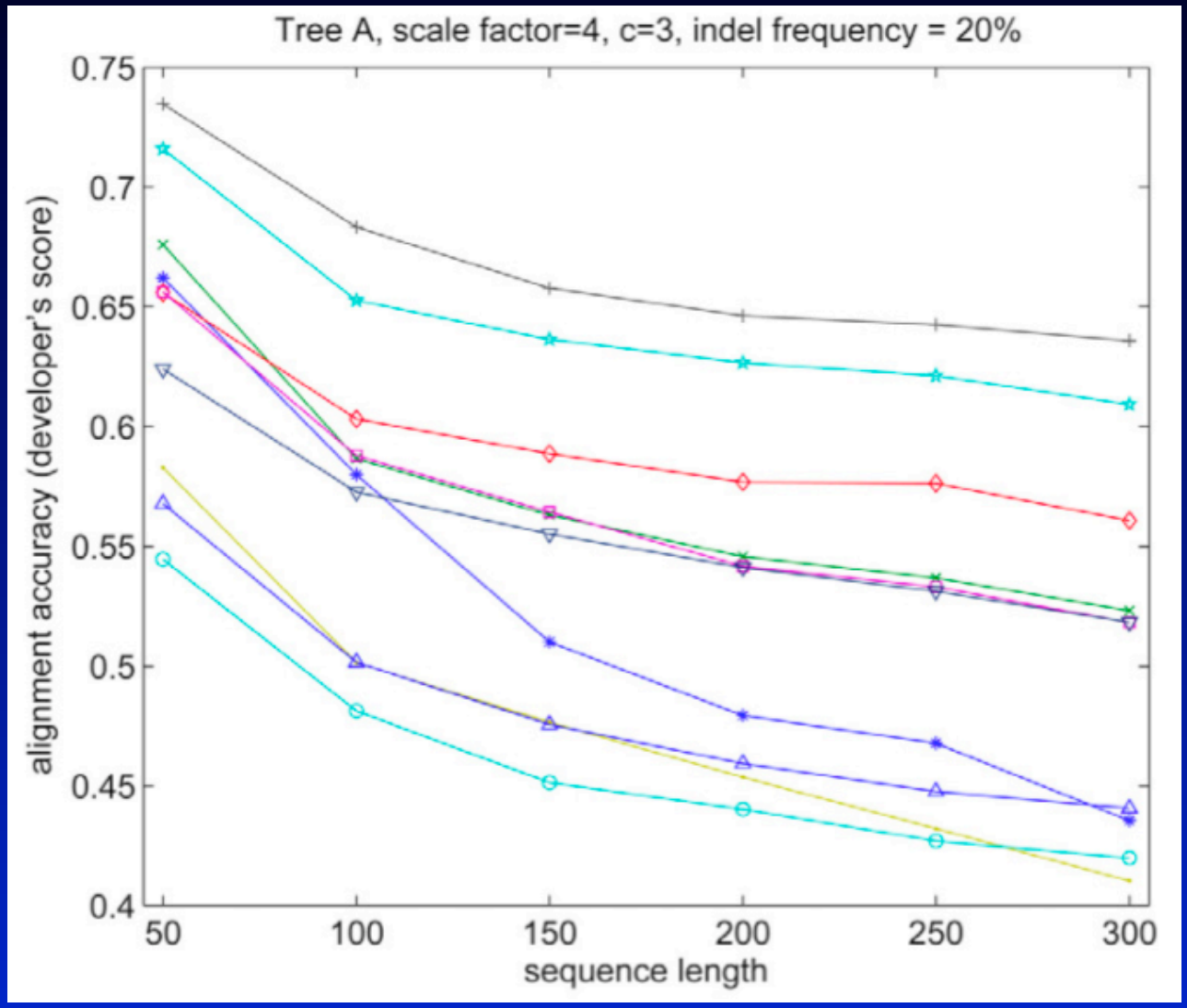
ClustalW

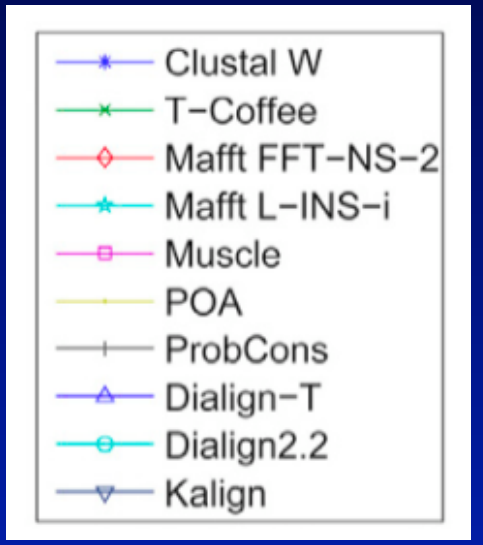
Poa

Performance on simulated data, few gaps



Performance on simulated data, many gaps





So which method should I choose?

- Performance depends on way of measuring and on nature of data set
- No single method performs best under all conditions (although mafft and ProbCons look quite good)
- To be on the safe side, you ought to check that results are robust to alignment uncertainty (try a number of methods, check conclusions on each alignment)
- Future perspectives: Bayesian techniques, alignment inferred along with rest of analysis, conclusions based on probability distribution over possible alignments.

Special purpose alignment programs

Nucleic Acids Research, 2003, Vol. 31, No. 13 ***3537–3539***
DOI: 10.1093/nar/gkg609

RevTrans: multiple alignment of coding DNA from aligned amino acid sequences

Rasmus Wernersson and Anders Gorm Pedersen*

Center for Biological Sequence Analysis, BioCentrum-DTU, Technical University of Denmark, Building 208, DK-2800, Lyngby, Denmark

- RevTrans: alignment of coding DNA based on information at protein level
- Codon-codon boundaries maintained

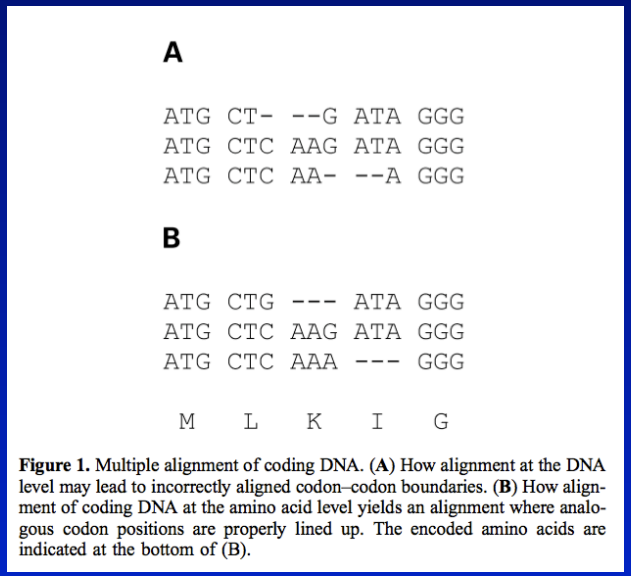


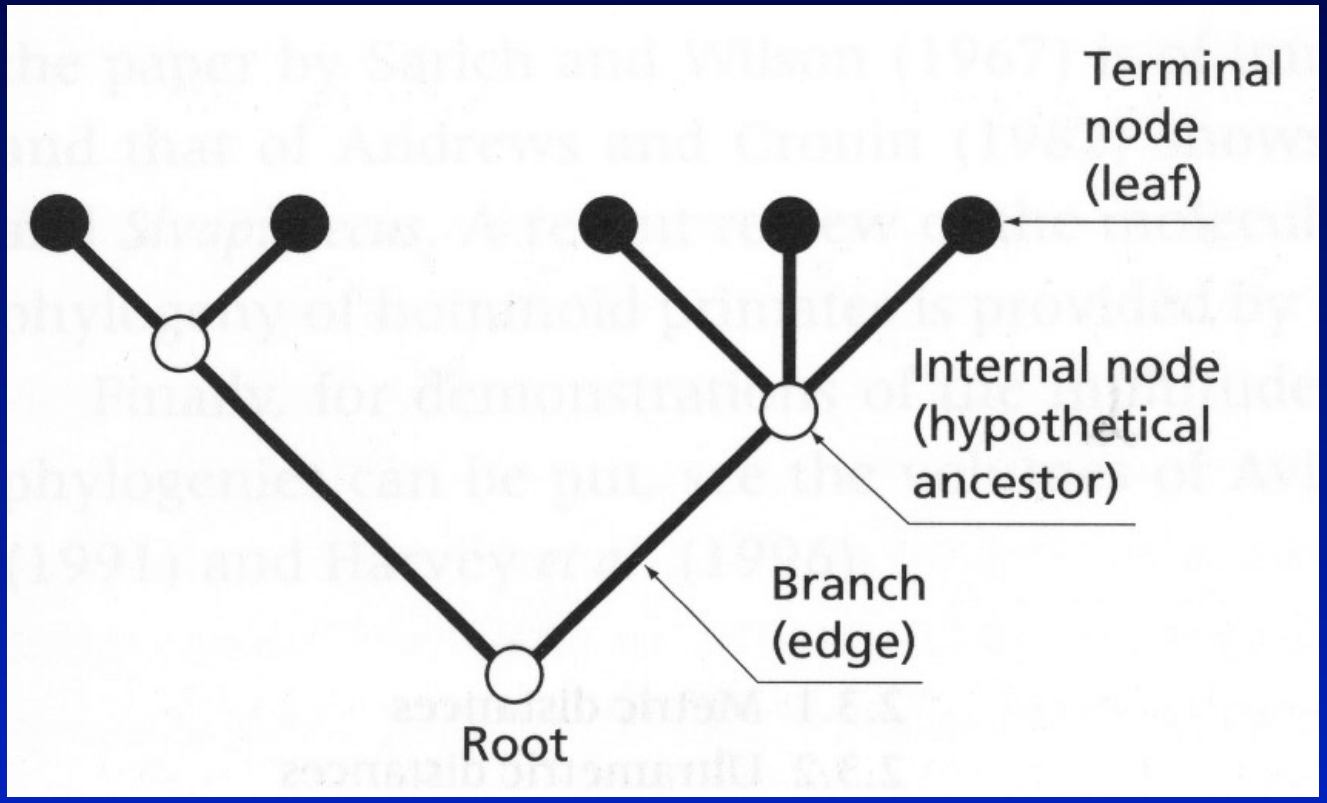
Figure 1. Multiple alignment of coding DNA. (**A**) How alignment at the DNA level may lead to incorrectly aligned codon–codon boundaries. (**B**) How alignment of coding DNA at the amino acid level yields an alignment where analogous codon positions are properly lined up. The encoded amino acids are indicated at the bottom of (B).

CENTER FOR BIOLOGICAL SEQUENCE ANALYSIS

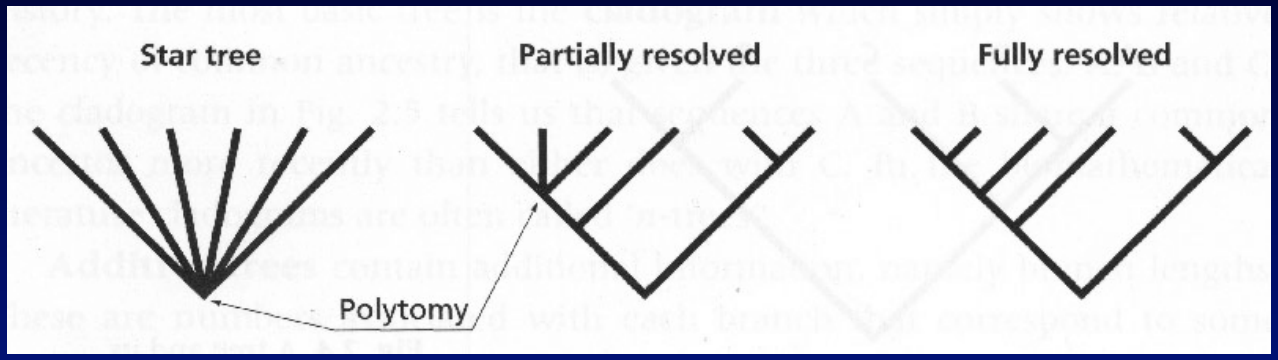
Phylogenetic Reconstruction: Distance Matrix Methods

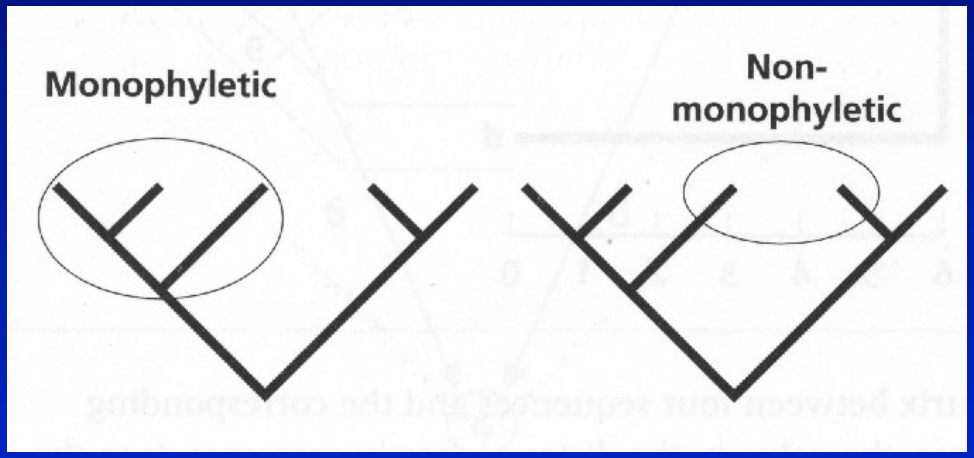
Anders Gorm Pedersen
Molecular Evolution Group
Center for Biological Sequence Analysis
Technical University of Denmark
gorm@cbs.dtu.dk

Trees: terminology

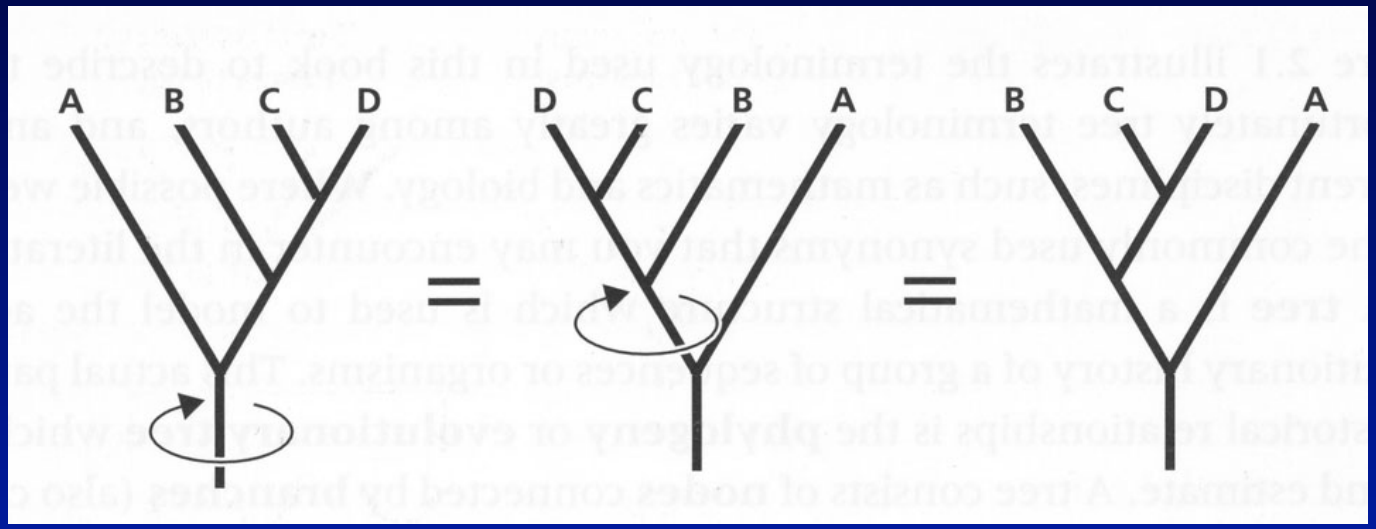


Trees: terminology



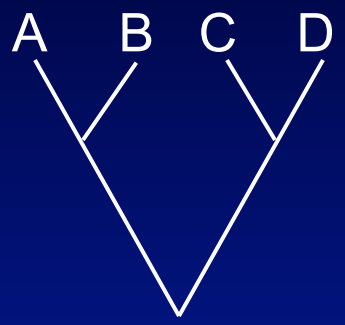


Trees: representations



Three different representations of the same tree

Trees: representation in computer files



((A , B) , (C , D));

Newick format:

- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

Newick format: named for seafood restaurant where standard was decided upon

Trees: representation in computer files

Newick format:

- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

Trees: representation in computer files

Newick format:

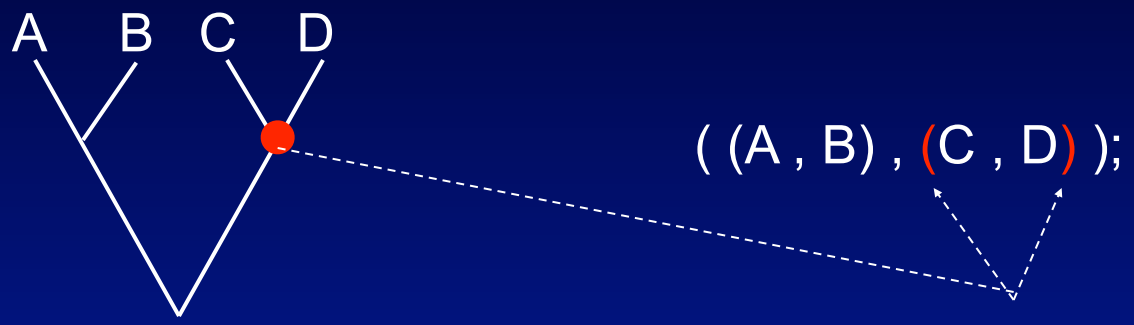
- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

Trees: representation in computer files

Newick format:

- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

Trees: representation in computer files



Newick format:

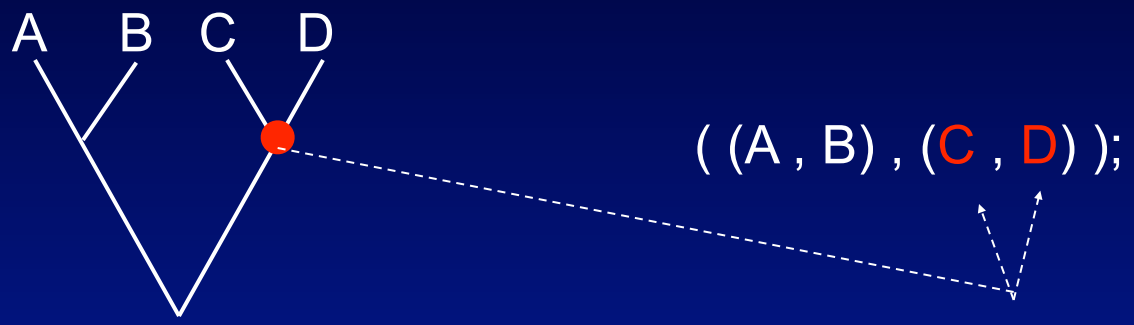
- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

Trees: representation in computer files

Newick format:

- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

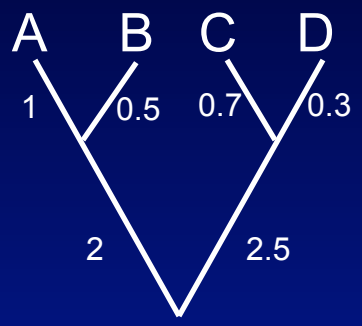
Trees: representation in computer files



Newick format:

- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

Trees: representation in computer files

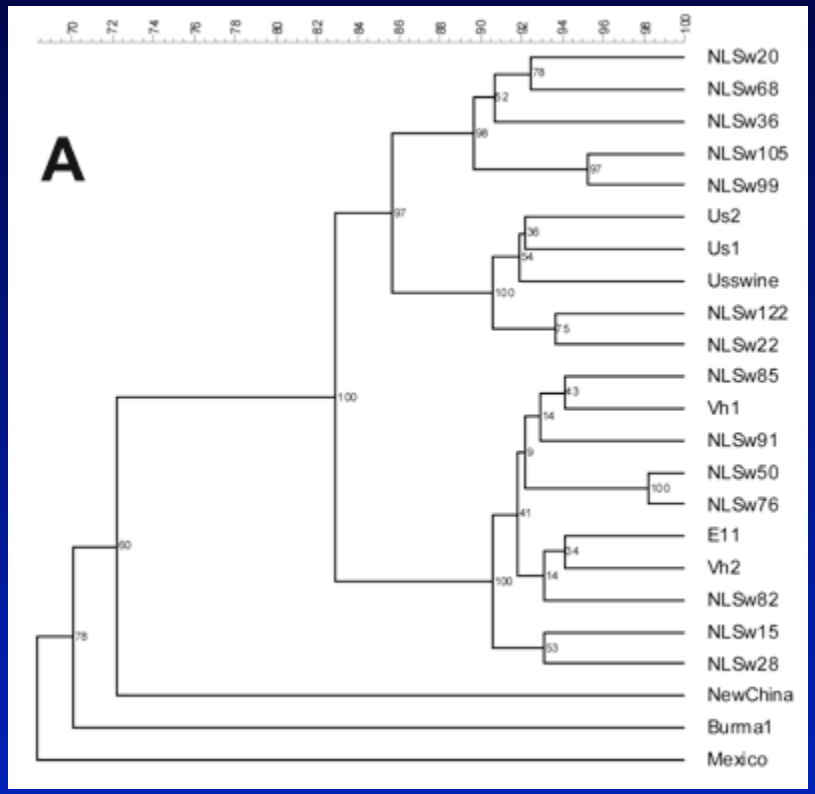


((A:1 , B:0.5) :2 , (C:0.7 , D:0.3) :2.5);

Newick format:

- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

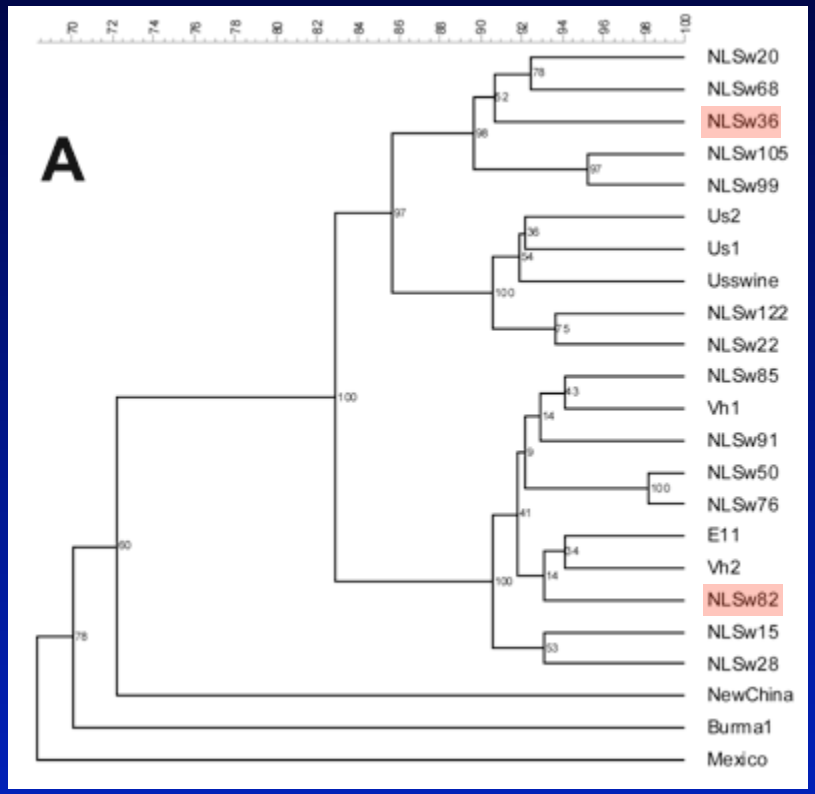
Trees: rooted vs. unrooted



Early → Late

- A rooted tree has a single node (the root) that represents a point in time that is earlier than any other node in the tree.
- A rooted tree has directionality (nodes can be ordered in terms of “earlier” or “later”).
- In the rooted tree, distance between two nodes is represented along the time-axis only (the second axis just helps spread out the leafs)

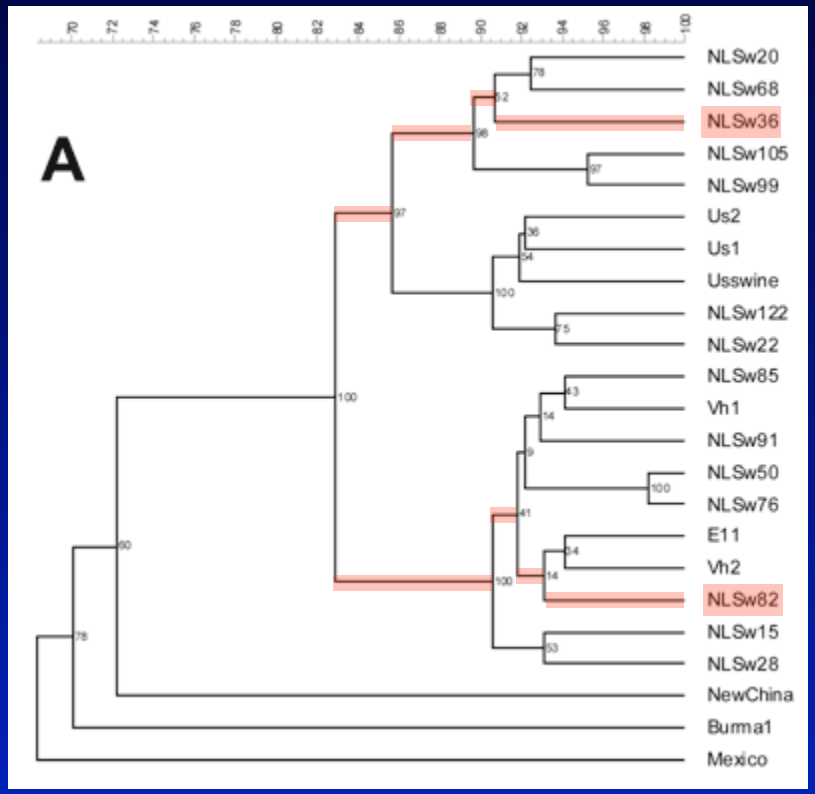
Trees: rooted vs. unrooted



Early → Late

- A rooted tree has a single node (the root) that represents a point in time that is earlier than any other node in the tree.
- A rooted tree has directionality (nodes can be ordered in terms of “earlier” or “later”).
- In the rooted tree, distance between two nodes is represented along the time-axis only (the second axis just helps spread out the leafs)

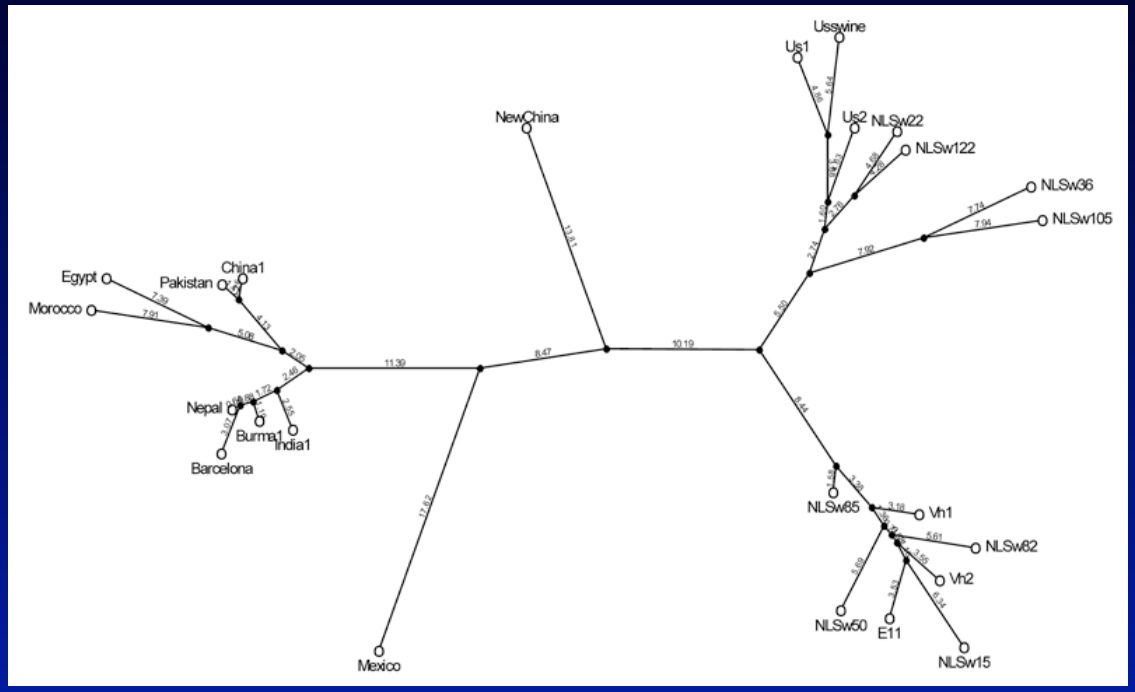
Trees: rooted vs. unrooted



Early → Late

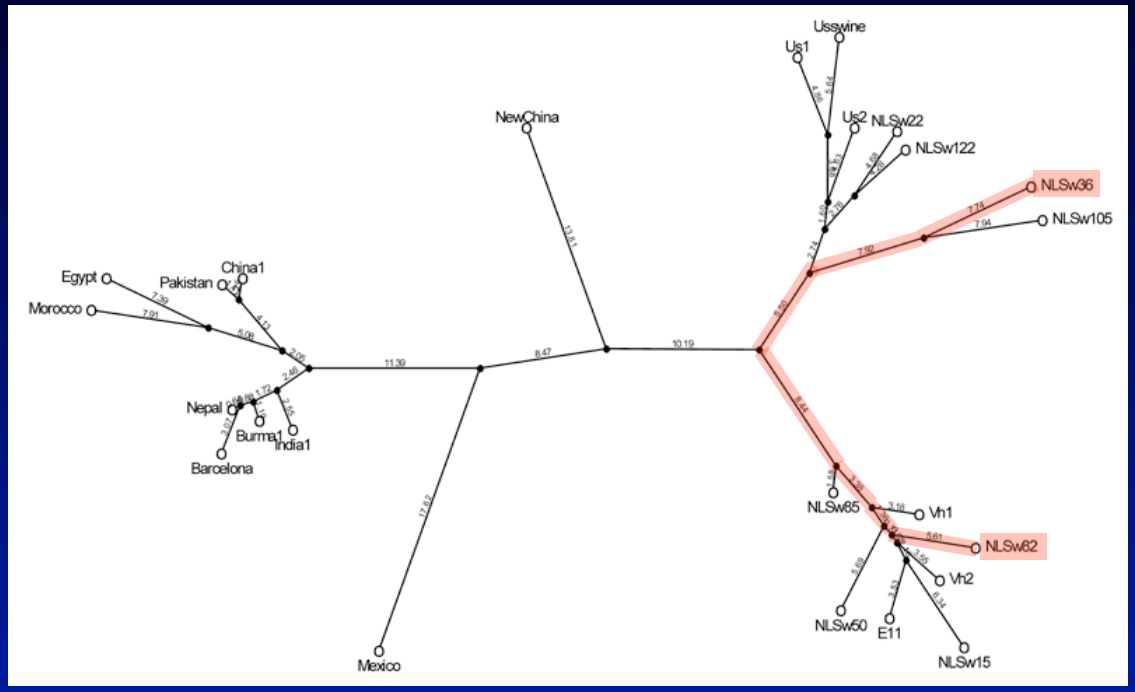
- A rooted tree has a single node (the root) that represents a point in time that is earlier than any other node in the tree.
- A rooted tree has directionality (nodes can be ordered in terms of “earlier” or “later”).
- In the rooted tree, distance between two nodes is represented along the time-axis only (the second axis just helps spread out the leafs)

Trees: rooted vs. unrooted



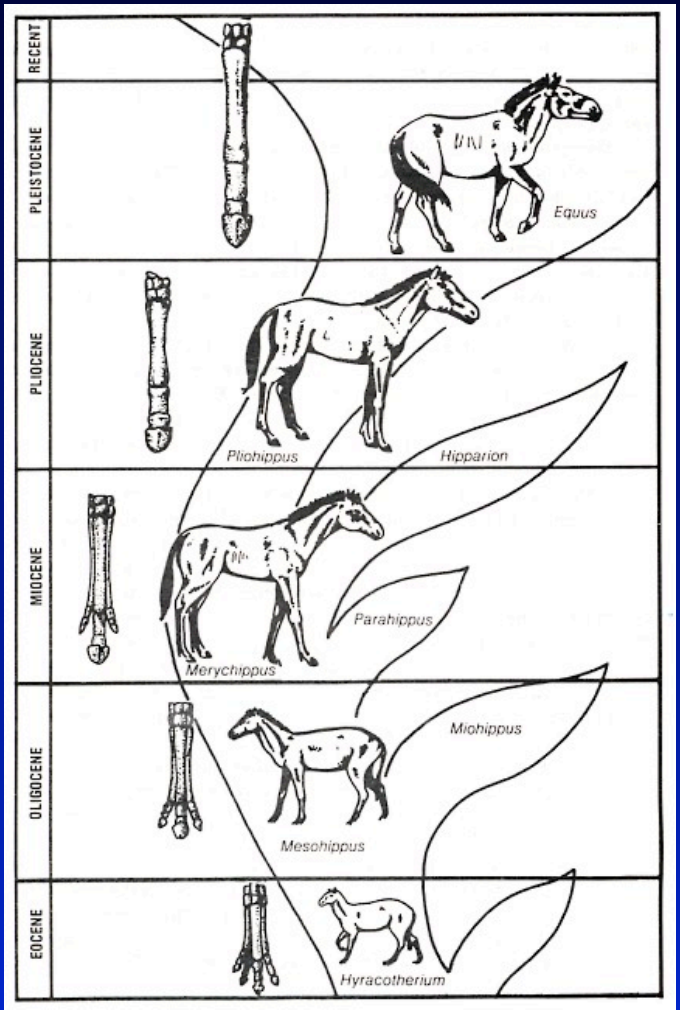
- In unrooted trees there is no directionality: we do not know if a node is earlier or later than another node
- Distance along branches directly represents node distance

Trees: rooted vs. unrooted

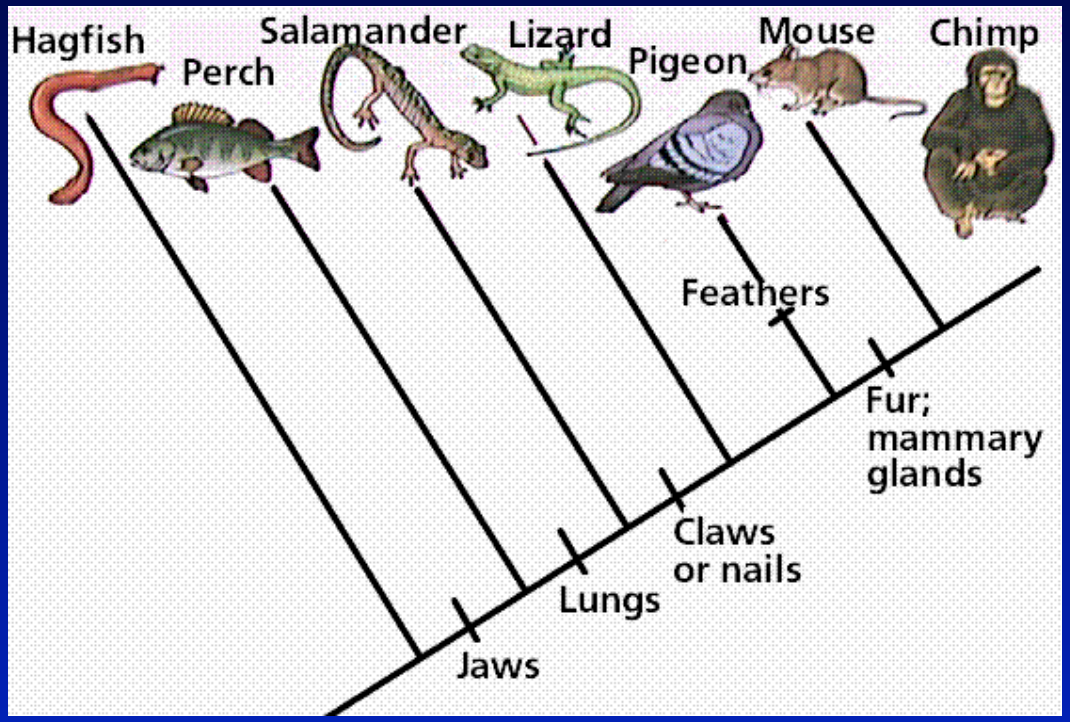


- In unrooted trees there is no directionality: we do not know if a node is earlier or later than another node
- Distance along branches directly represents node distance

Reconstructing a tree using non-contemporaneous data



Reconstructing a tree using present-day data



Data: molecular phylogeny

- DNA sequences
  - genomic DNA
  - mitochondrial DNA
  - chloroplast DNA
- Protein sequences
- Restriction site polymorphisms
- DNA/DNA hybridization
- Immunological cross-reaction

Morphology vs. molecular data

African white-backed vulture
(old world vulture)

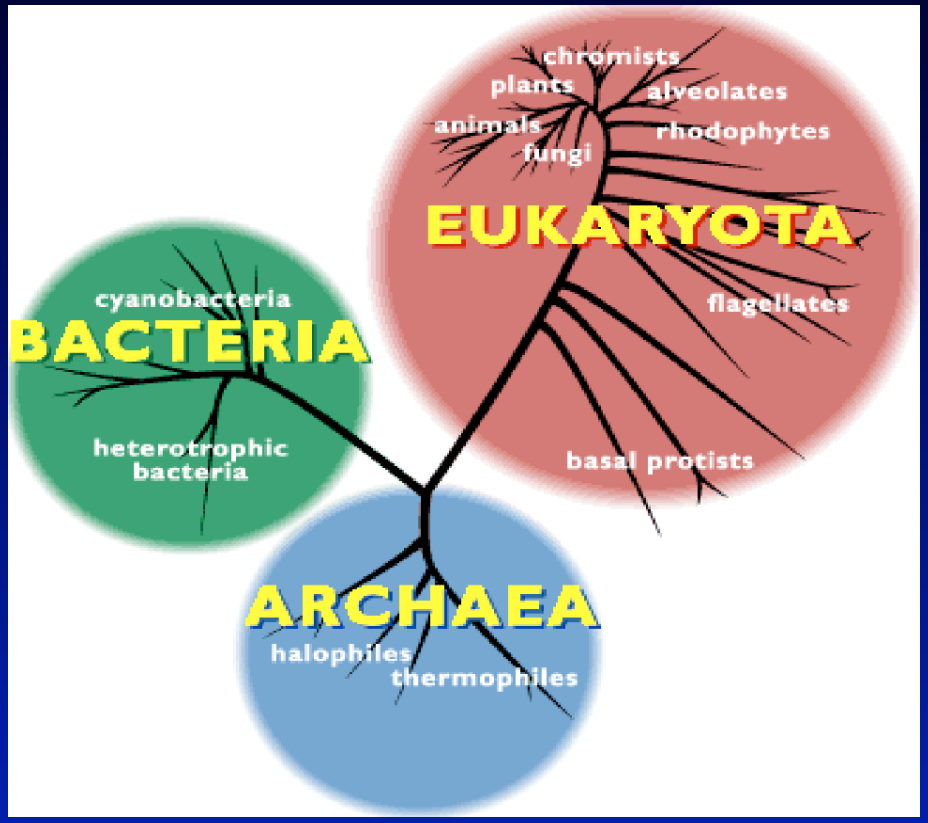
Andean condor
(new world vulture)

New and old world vultures seem to be closely related based on morphology.

Molecular data indicates that old world vultures are related to birds of prey (falcons, hawks, etc.) while new world vultures are more closely related to storks

Similar features presumably the result of convergent evolution

Molecular data: single-celled organisms



Molecular data useful for analyzing single-celled organisms (which have only few prominent morphological features).

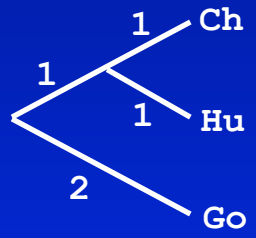
Distance Matrix Methods


```
Gorilla    :  ACGTCGTA
Human      :  ACGTTCCT
Chimpanzee :  ACGTTTCG
```

1. Construct multiple alignment of sequences

	Go	Hu	Ch
Go	-	4	4
Hu		-	2
Ch			-

2. Construct table listing all pairwise differences (distance matrix)

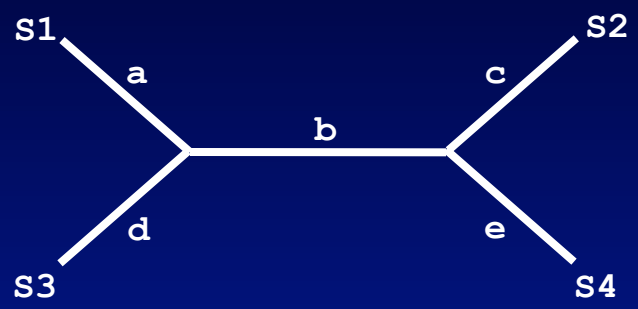


3. Construct tree from pairwise distances

Finding Optimal Branch Lengths

	S_1	S_2	S_3	S_4
S_1	-	12	13	14
S_2		-	23	24
S_3			-	34
S_4				-

Observed distance

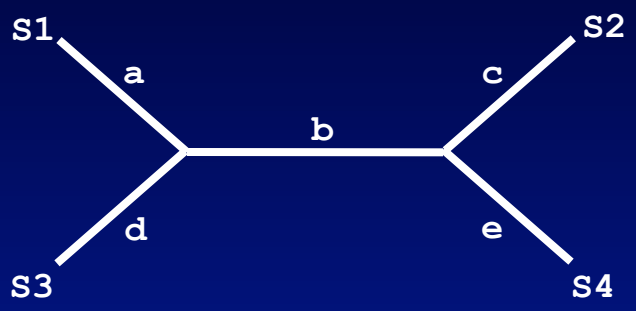


Distance along tree

Goal:

$$D_{12} \approx d_{12} = a + b + c$$
$$D_{13} \approx d_{13} = a + d$$
$$D_{14} \approx d_{14} = a + b + e$$
$$D_{23} \approx d_{23} = d + b + c$$
$$D_{24} \approx d_{24} = c + e$$
$$D_{34} \approx d_{34} = d + b + e$$

Optimal Branch Lengths: Least Squares



Distance along tree

Goal:

$$D_{12} \approx d_{12} = a + b + c$$
$$D_{13} \approx d_{13} = a + d$$
$$D_{14} \approx d_{14} = a + b + e$$
$$D_{23} \approx d_{23} = d + b + c$$
$$D_{24} \approx d_{24} = c + e$$
$$D_{34} \approx d_{34} = d + b + e$$

- Fit between given tree and observed distances can be expressed as “sum of squared differences”:

$$Q = \sum_{j>i} (D_{ij} - d_{ij})^2$$

- Find branch lengths that minimize Q - this is the optimal set of branch lengths for this tree.

Least Squares Optimality Criterion

- Search through all (or many) tree topologies
- For each investigated tree, find best branch lengths using least squares criterion
- Among all investigated trees, the best tree is the one with the smallest sum of squared errors.

Exhaustive search impossible for large data sets

No. taxa	No. trees
3	1
4	3
5	15
6	105
7	945
8	10,395
9	135,135
10	2,027,025
11	34,459,425
12	654,729,075
13	13,749,310,575
14	316,234,143,225
15	7,905,853,580,625

Heuristic search

1. Construct initial tree; determine sum of squares
2. Construct set of “neighboring trees” by making small rearrangements of initial tree; determine sum of squares for each neighbor
3. If any of the neighboring trees are better than the initial tree, then select it/them and use as starting point for new round of rearrangements. (Possibly several neighbors are equally good)
4. Repeat steps 2+3 until you have found a tree that is better than all of its neighbors.
5. This tree is a “local optimum” (not necessarily a global optimum!)

Clustering Algorithms

- Starting point: Distance matrix
- Cluster least different pair of sequences:
- Repeat until all nodes are linked
- Results in only one tree, there is no measure of tree-goodness.

Neighbor Joining Algorithm

- For each tip compute $u_i = \Sigma_j D_{ij}/(n-2)$

  (this is essentially the average distance to all other tips, except the denominator is n-2 instead of n)

- Find the pair of tips, i and j, where $D_{ij}-u_i-u_j$ is smallest

- Connect the tips i and j, forming a new ancestral node. The branch lengths from the ancestral node to i and j are:

$$v_i = 0.5\ D_{ij} + 0.5\ (u_i-u_j)$$

$$v_j = 0.5\ D_{ij} + 0.5\ (u_j-u_i)$$

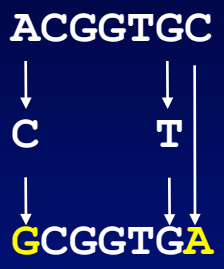
- Update the distance matrix: Compute distance between new node and each remaining tip as follows:

$$D_{ij,k} = (D_{ik}+D_{jk}-D_{ij})/2$$

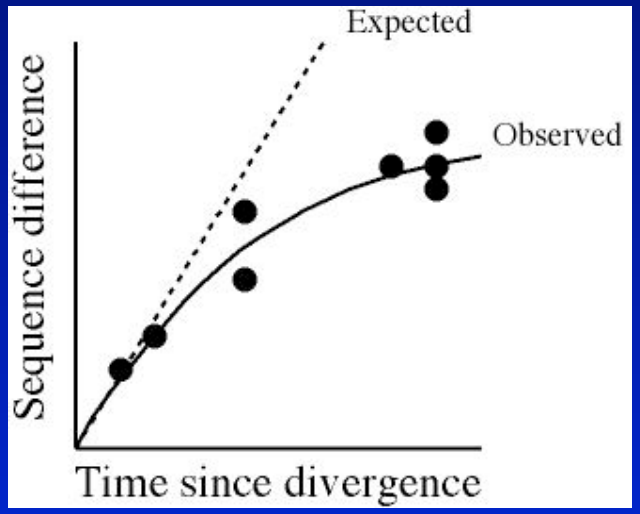
- Replace tips i and j by the new node which is now treated as a tip

- Repeat until only two nodes remain.

Superimposed Substitutions



- Actual number of evolutionary events: 5
- Observed number of differences: 2



- Distance is (almost) always underestimated

Model-based correction for superimposed substitutions

- Goal: try to infer the real number of evolutionary events (the real distance) based on

1. Observed data (sequence alignment)

2. A model of how evolution occurs

Jukes and Cantor Model

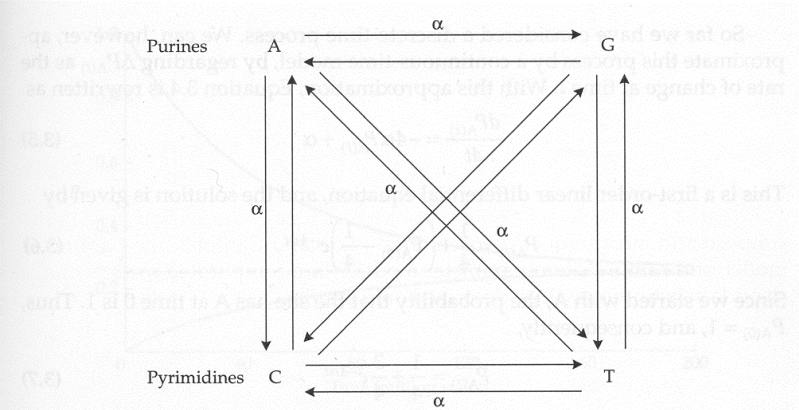


FIGURE 3.1 One-parameter model of nucleotide substitution. The rate of substitution in each direction is α.

	A	C	G	T
A	-3α	α	α	α
C	α	-3α	α	α
G	α	α	-3α	α
T	α	α	α	-3α

- Four nucleotides assumed to be equally frequent (f=0.25)
- All 12 substitution rates assumed to be equal
- Under this model the corrected distance is:

$$D_{JC} = -0.75 \times \ln(1-1.33 \times D_{OBS})$$

- For instance:

$$D_{OBS}=0.43 \Rightarrow D_{JC}=0.64$$

Other models of evolution

TABLE 3.1 Models of nucleotide substitution

O\S[a]	A	T	C	G
a. Two-parameter model (Kimura 1980)				
A	$1-\alpha-2\beta$	β	β	α
T	β	$1-\alpha-2\beta$	α	β
C	β	α	$1-\alpha-2\beta$	β
G	α	β	β	$1-\alpha-2\beta$
b. Four-parameter model (Blaisdell 1985)				
A	$1-\alpha-2\gamma$	γ	γ	α
T	δ	$1-\alpha-2\delta$	α	δ
C	δ	β	$1-\beta-2\delta$	δ
G	β	γ	γ	$1-\beta-2\gamma$
c. Six-parameter model (Kimura 1981a)				
A	$1-2\alpha-\gamma$	γ	α	α
T	δ	$1-2\alpha-\delta$	α	α
C	β	β	$1-2\beta-\epsilon$	ϵ
G	β	β	ξ	$1-2\beta-\xi$
d. Nine-parameter model				
A	$1-g_T\beta_1-g_C\gamma_1-g_G\alpha_1$	$g_T\beta_1$	$g_C\gamma_1$	$g_G\alpha_1$
T	$g_A\beta_1$	$1-g_A\beta_1-g_C\alpha_2-g_G\gamma_2$	$g_C\alpha_2$	$g_G\gamma_2$
C	$g_A\gamma_1$	$g_T\alpha_2$	$1-g_A\gamma_1-g_T\alpha_2-g_G\beta_2$	$g_G\beta_2$
G	$g_A\alpha_1$	$g_T\gamma_2$	$g_C\beta_2$	$1-g_A\alpha_1-g_T\gamma_2-g_C\beta_2$
e. General model				
A	$1-\alpha_{12}-\alpha_{13}-\alpha_{14}$	α_{12}	α_{13}	α_{14}
T	α_{21}	$1-\alpha_{21}-\alpha_{23}-\alpha_{24}$	α_{23}	α_{24}
C	α_{31}	α_{32}	$1-\alpha_{31}-\alpha_{32}-\alpha_{34}$	α_{34}
G	α_{41}	α_{42}	α_{43}	$1-\alpha_{41}-\alpha_{42}-\alpha_{43}$

[a]O, Original nucleotide; S, substitute nucleotide.